Figure 1A

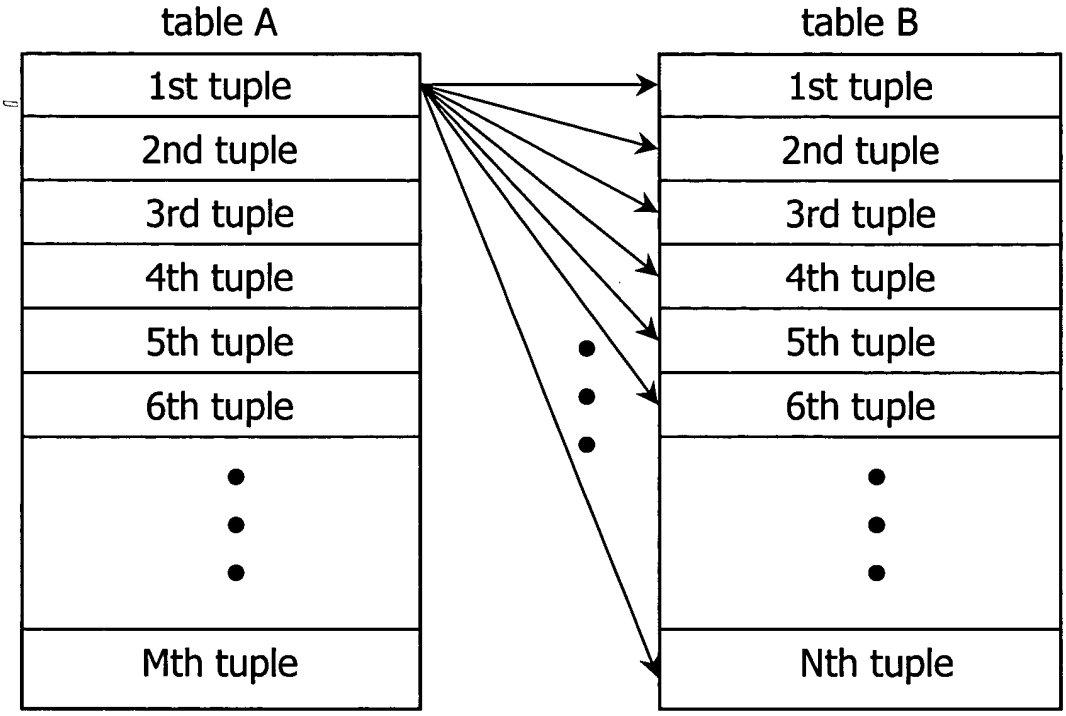


Figure 1B

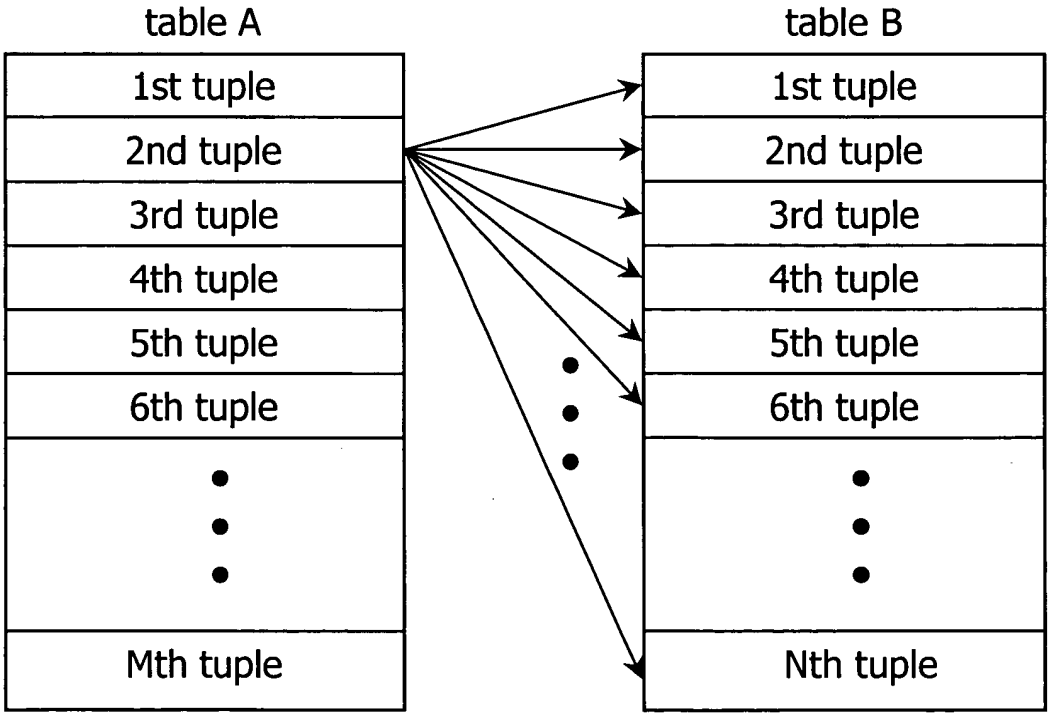


Figure 2

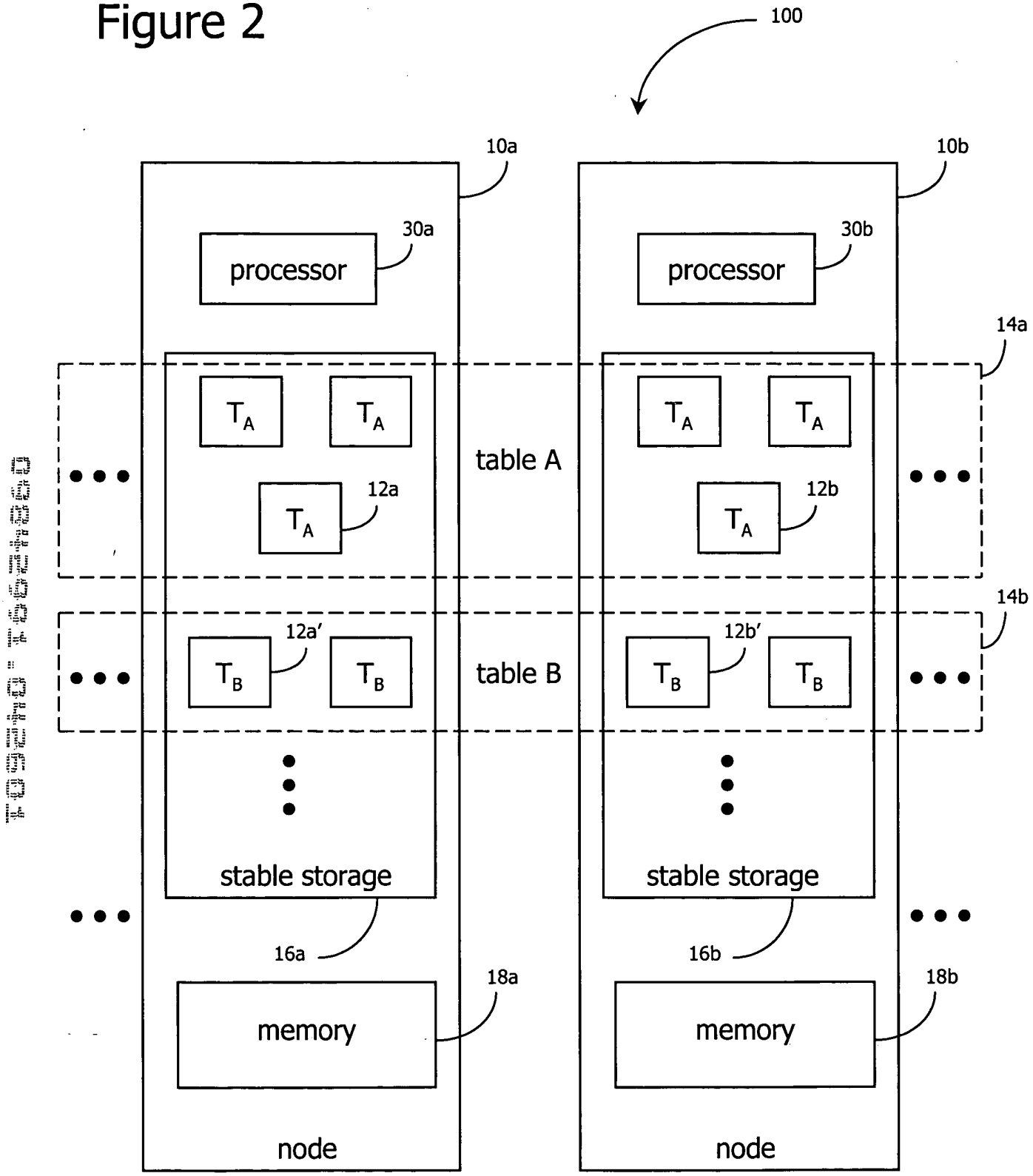

100
10a
10b
30a
processor
30b
processor
14a
T_A
T_A
T_A
table A
12a
T_A
T_A
T_A
12b
T_A
14b
12a'
T_B
T_B
table B
12b'
T_B
T_B
stable storage
stable storage
16a
16b
18a
memory
18b
memory
node
node

Figure 3

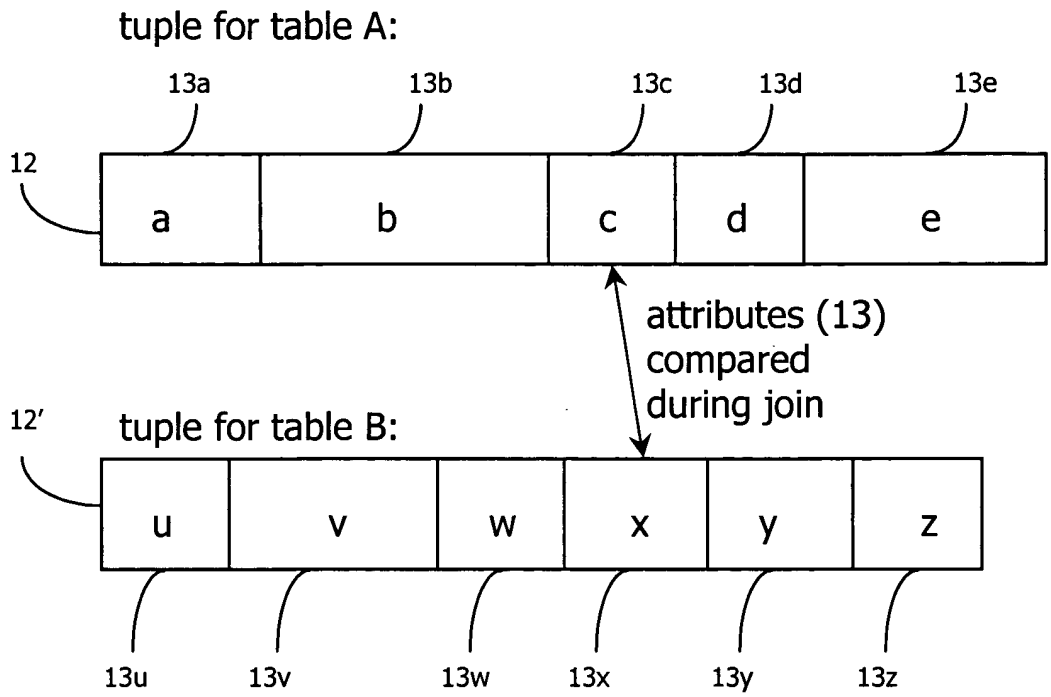

tuple for table A:
13a
13b
13c
13d
13e
12
a
b
c
d
e
attributes (13)
compared
during join
12'
tuple for table B:
u
v
w
x
y
z
13u
13v
13w
13x
13y
13z

Figure 4

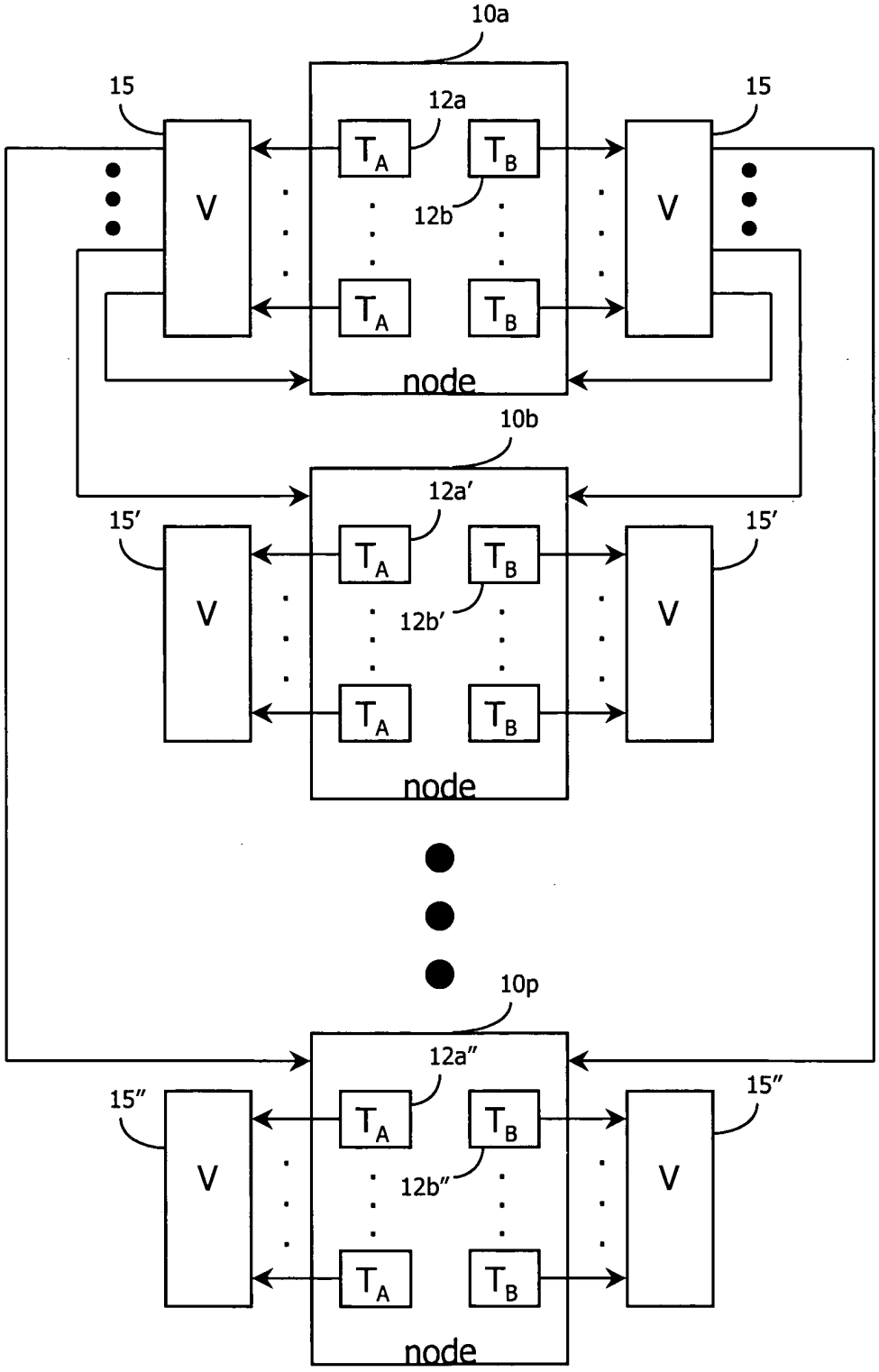

10a
15
12a
15
V
T_A
T_B
12b
V
T_A
T_B
node
10b
12a'
15'
15'
V
T_A
T_B
12b'
V
T_A
T_B
node
10p
12a"
15"
15"
V
T_A
T_B
12b"
V
T_A
T_B
node

Figure 5

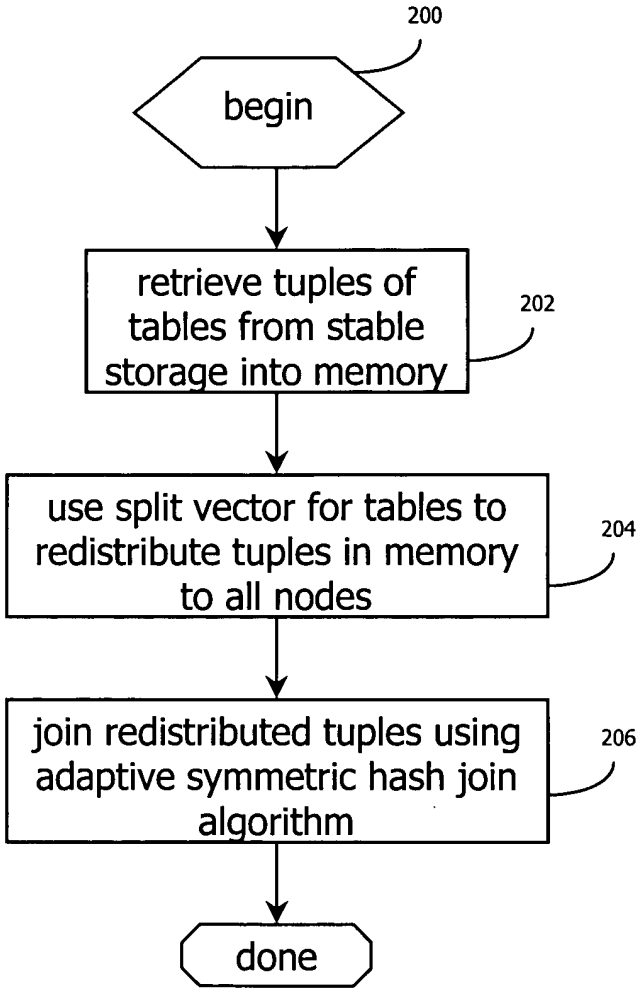

200
begin
retrieve tuples of tables from stable storage into memory
202
use split vector for tables to redistribute tuples in memory to all nodes
204
join redistributed tuples using adaptive symmetric hash join algorithm
206
done

Figure 6

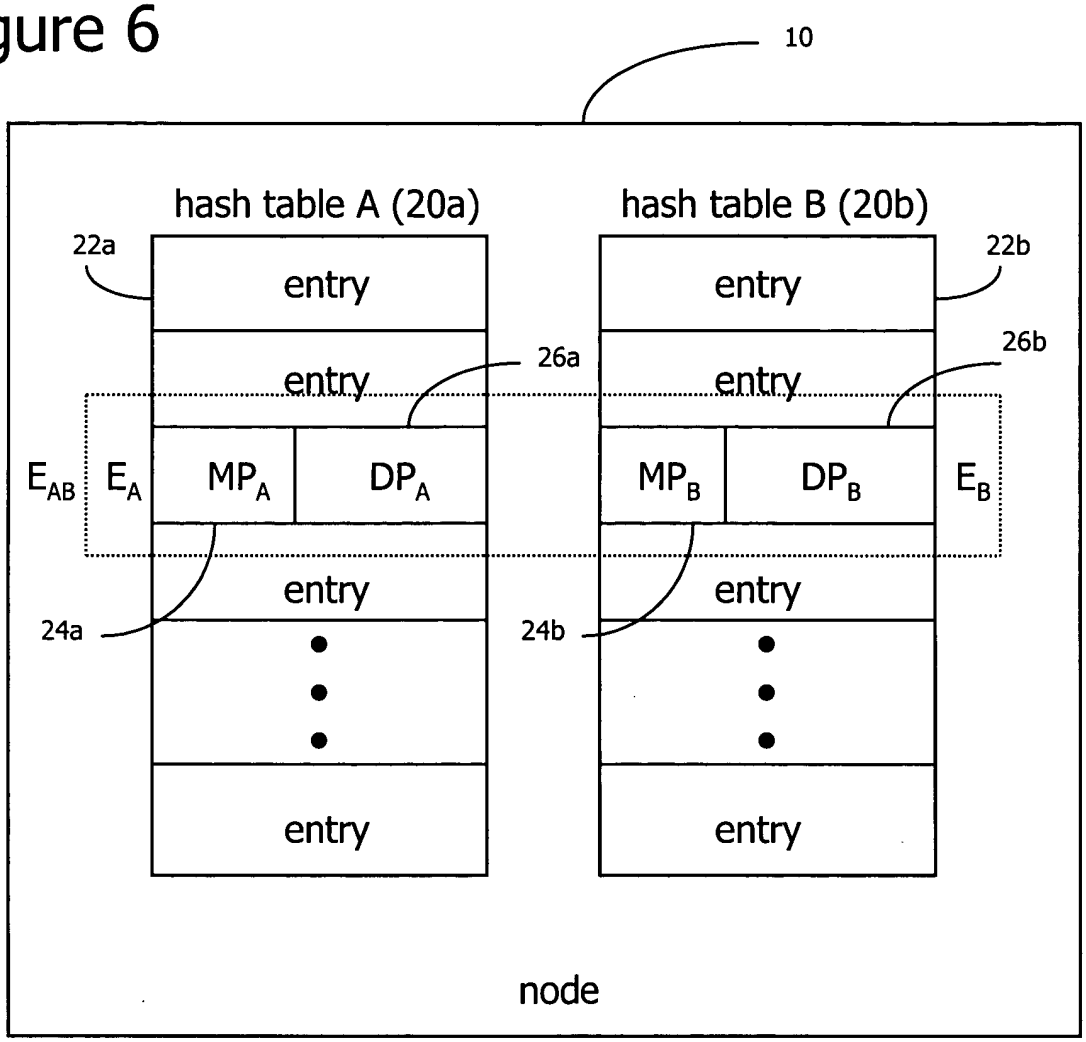

10
hash table A (20a)
hash table B (20b)
22a
22b
entry
entry
entry
entry
26a
26b
E_{AB}
E_A
MP_A
DP_A
MP_B
DP_B
E_B
entry
entry
24a
24b
entry
entry
node

Figure 7

first stage

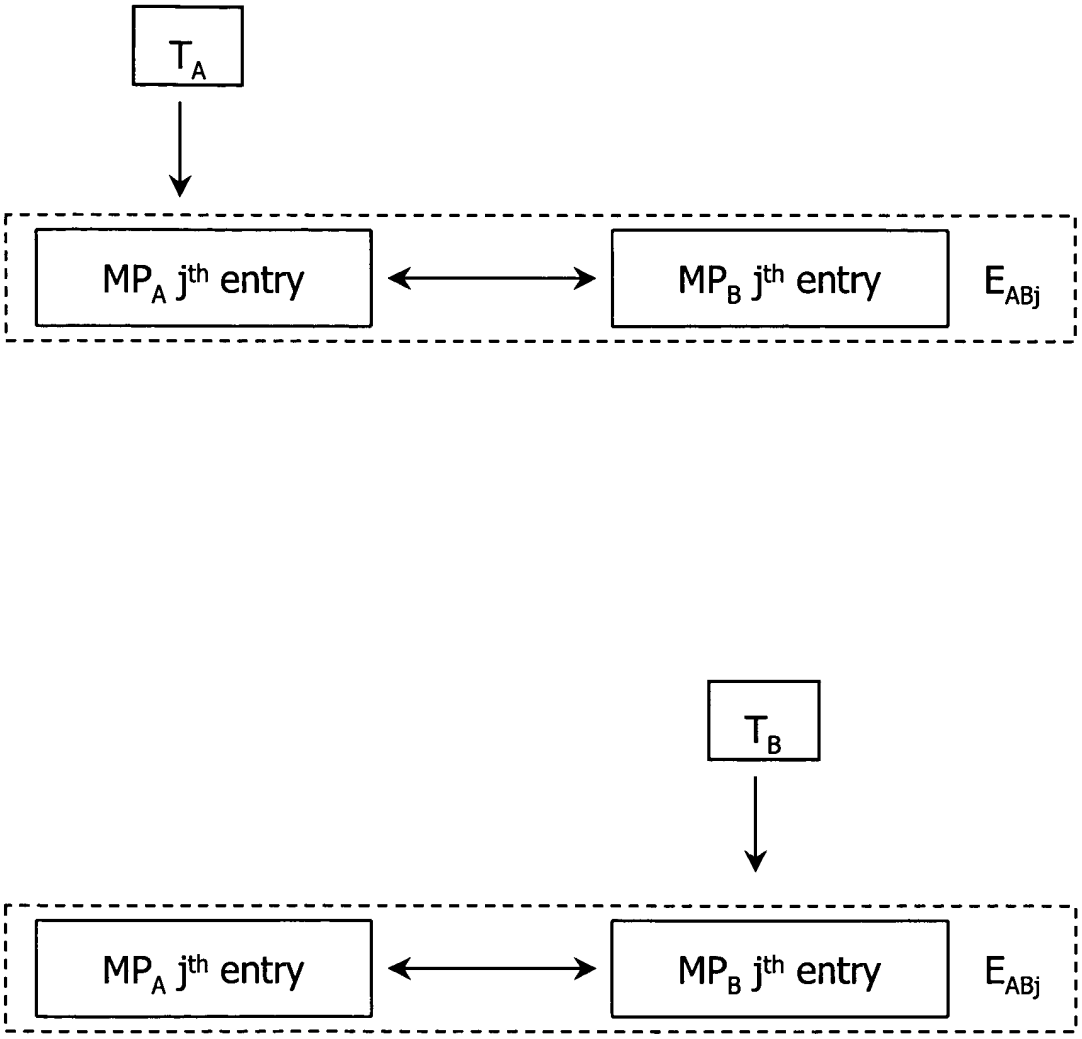
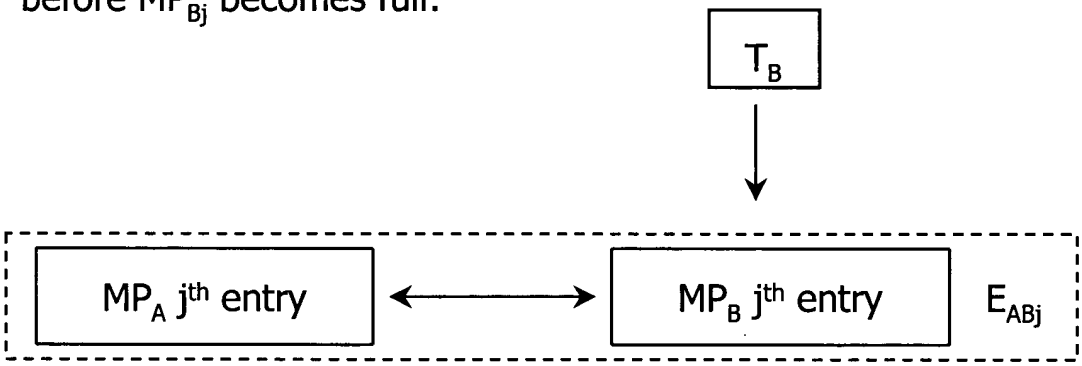


Figure 8A

second stage

MP_{Aj} became full before MP_{Bj} became full:

before MP_{Bj} becomes full:



after MP_{Bj} becomes full:

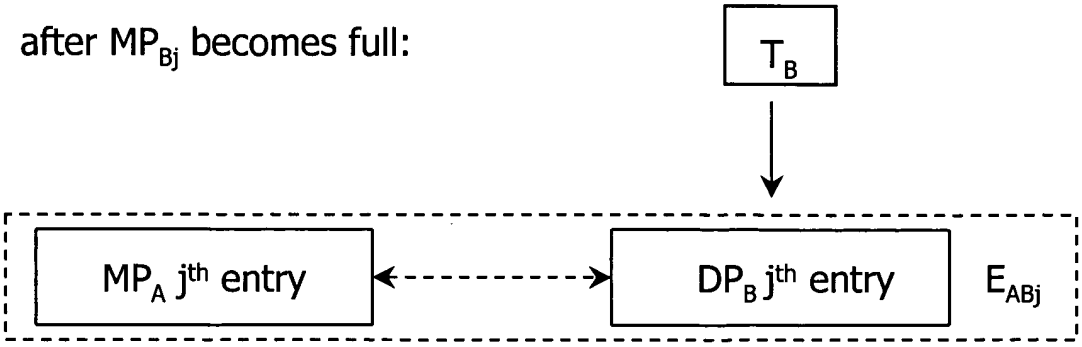
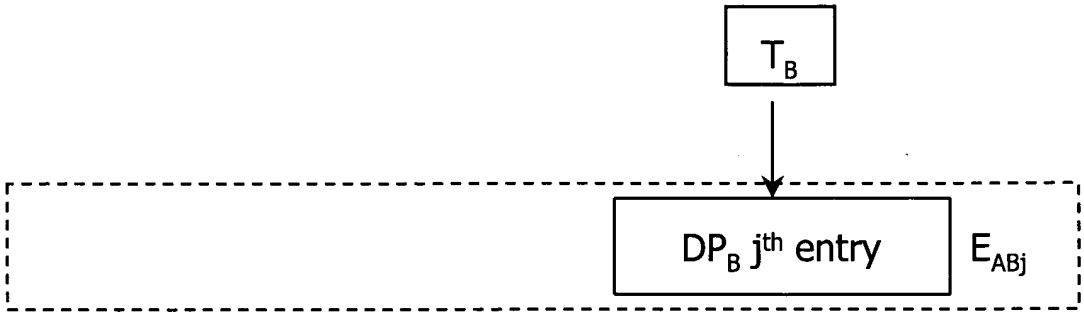


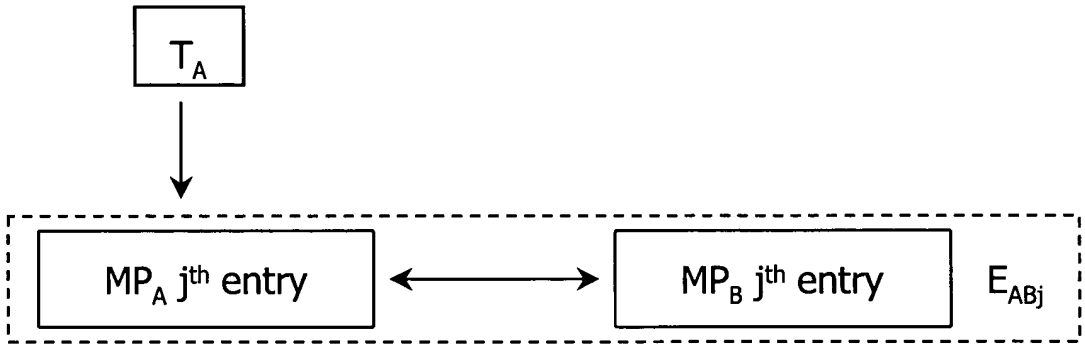
Figure 8B

second stage (cont.)

MP_{Bj} became full before MP_{Aj} became full:



before MP_{Aj} becomes full:



after MP_{Aj} becomes full:

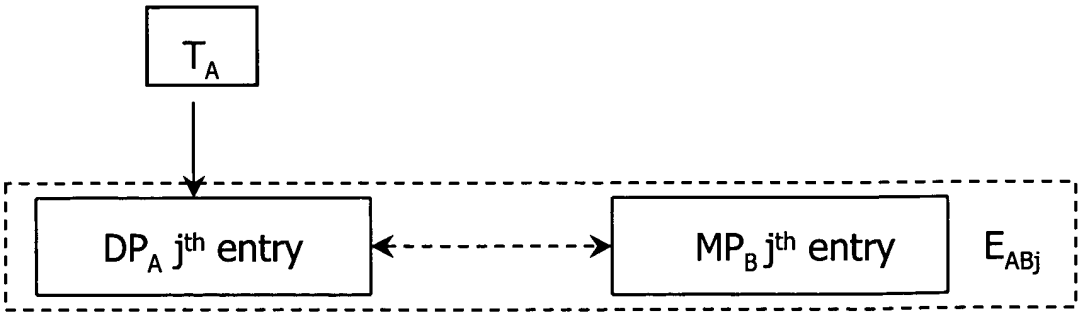
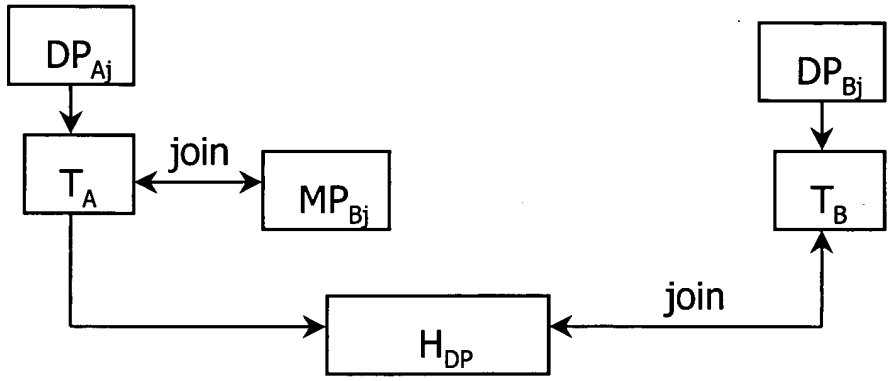
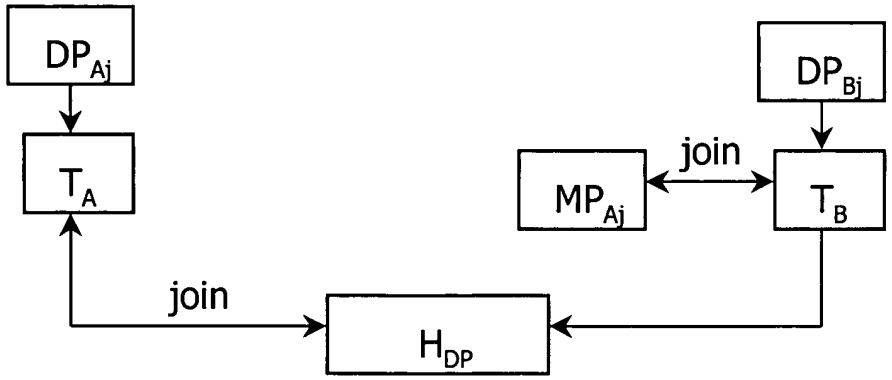


Figure 9

third stage

MP_{Aj} became full before MP_{Bj} became full:



MP_{Bj} became full before MP_{Aj} became full: